VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Mechanizmy řízení robotického auta NXP (FREESCALE)

Driving Mechanizms of Robotic Car NXP (FREESCALE)

2017 Tomáš Paleček

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Tomáš Paleček**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Mechanizmy řízení robotického auta NXP (FREESCALE)
Driving Mechanizms of Robotic Car NXP (FREESCALE)

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je vytvořit software pro ovládání robotického auta FREESCALE s kitem FRDMZKL25Z. Software umožní účast na soutěži NXP Cup a připojení nadřazeného řídícího modulu Raspberry Pi.

Celý software bude sestávat ze tří hlavních částí:
1. Část pro ovládání hardwaru auta napsaná v C++ pro kit FRDMZKL25Z.
2. Část pro řízení jízdy auta po trati vyznačené černými okrajovými čárami (C++ pro kit FRDMZKL25Z).
3. Část pro komunikaci s modulem Raspberry Pi (kit FRDMZKL25Z i Raspberry Pi) umožňující sofistikovanější mechanizmus řízení.

Práce bude obsahovat:
1. Přehled používaných metod a algoritmů.
2. Implementaci výše popsané funkcionality.
3. Experimenty a vyhodnocení výsledků.
4. Programátorskou dokumentaci řešení s využitím diagramů jazyka UML.

Seznam doporučené odborné literatury:

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (Gang of Four): Návrh programů pomocí vzorů. Grada. Praha 2003. ISBN 8024703025
[2] DARWIN, Ian F. Java cookbook. 2nd ed. Sebastopol, CA: O'Reilly, c2004, xxiv, 829 p. ISBN 05-960-0701-9. Dostupné z: http://it-ebooks.info/book/2249/

Dále dle pokynů vedoucího práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Ježek, Ph.D.**

Datum zadání: 01.09.2016

Datum odevzdání: 28.04.2017

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 28. dubna 2017 .....

Rád bych tímto poděkoval svému vedoucímu bakalářské práce, panu Ing. Davidu Ježkovi, Ph.D., za věnovaný čas a skvělé vedení. Dále bych rád poděkoval panu Ing. Petru Olivkovi, Ph.D. za cenné rady a podporu při vývoji projektu.

Abstrakt

Tato bakalářská práce se věnuje metodám řízení robotického auta NXP pomocí jednořádkové kamery v prostředí závodní dráhy ohraničené krajnicemi. Za tímto účelem byly navrženy vlastní nebo převzaty existující metody filtrace obrazu, metody řízení auta a způsoby komunikace robotického auta s dalšími zařízeními. Při filtraci obrazu byly použity metody běžně používané při zpracování obrazu a některé metody řízení robotického auta byly inspirovány vlastnostmi skutečných automobilů. Součástí cílů práce byla účast na soutěži NXP Cup, kde mezi sebou robotické auta řešící stejnou problematiku závodí.

Klíčová slova: NXP, Freescale, robotické auto, autonomní řízení, filtrace obrazu

Abstract

This bachelor work deals with methods of driving of robotic car NXP using line camera on racetrack bounded by outboards. In the work own or existing ones methods of image processing, car driving and methods of communication between robotic car and external devices were used. For image filtration commonly used methods were used and selected methods of robotic car driving were inspired by properties of standard cars. One of the aims of the work was participation on the challenge NXP Cup where the robotic cars dealing with the same issue are challenging.

Key Words: NXP, Freescale, robotic car, autonomous driving, image filtering

Obsah

Seznam použitých zkratek a symbolů

ADC	–	Analog-to-digital converter
CCD	–	Charge-coupled device
DIP	–	Dual Inline Package
GPIO	–	General-purpose input/output
PWM	–	Pulse-width modulation
TCP	–	Transmission Control Protocol
UDP	–	User Datagram Protocol
USB CDC	–	Universal Serial Bus communications device class

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Tato práce vznikla na základě týmového projektu s účelem vytvoření programu autonomního řízení pro robotické auto od firmy NXP® Semiconductors N.V.. Jedním z cílů vytvoření programu bylo umožnění účasti na mezinárodní soutěži NXP Cup roku 2017, pořádanou firmou NXP. Soutěž je pořádána formou závodu robotických aut firmy NXP (dále jen aut), jehož účastníky jsou zástupci (týmy) z vysokých a středních škol. Program pro autonomní řízení auta byl navržen s ohledem na požadavky a pravidla této soutěže ročníku 2017. Jedním ze základních požadavků bylo, aby auto za pomocí jednořádkové kamery a základního vývojového kitu, projelo drahou ohraničenou dvěmi krajnicemi - obdobně jako v reálném světě auto jezdí po silnici tmavé barvy ohraničené dvěmi bílými krajnicemi. Práce byla vyvíjena v programovacích jazycích C, C++ a Java. Byl využit verzovací systém Git ke správě verzí a projektový management byl zajištěn za pomocí systému Redmine. Pro účely bakalářské práce, byla výbava auta a funkcionalita rozšířena nad rámec požadavků a pravidel soutěže NXP Cup. V rámci vývoje byl pro projekt dodán panem Ing. Davidem Ježkem, Ph.D. podpůrný software pro ladění programu a existující program pro učení pomocí neuronových sítí pro experimenty s Raspberry Pi. Dále byl pro projekt panem Ing. Petrem Olivkou, Ph.D. dodán interface pro ovládání hardware za účelem sjednocení přístupu k hardware všech NXP aut vlastněných univerzitou.

2 Vybavení

2.1 Hardware aut

Pro práci na tomto projektu byly použity dvě verze robotického auta. Jedno robotické auto bylo vybaveno vybavením striktně povoleným pravidly soutěže NXP Cup 2017 a druhé auto, pro účely bakalářské práce, bylo vybaveno rozšířenými periferiemi.

2.1.1 Soutěžní verze auta

Stručná specifikace soutěžního auta (obrázek 1a):

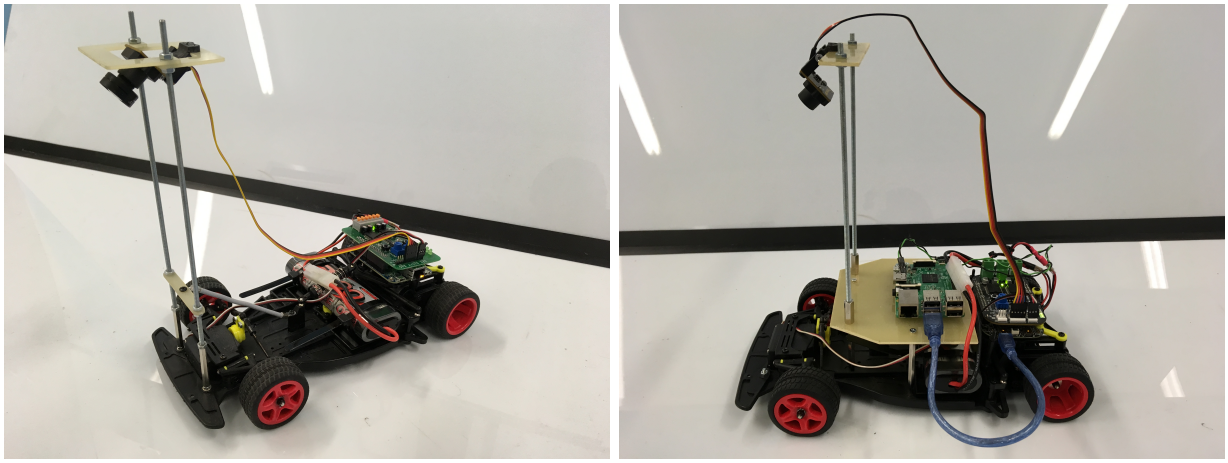
- skeleton auta
  - rozchod zadních i předních kol 160 mm
  - rozvor náprav 200 mm
  - výška umístění kamery od země 270 mm
  - vzdálenost umístění vztyčných ramen kamery 40 mm za přední nápravou
  - maximální úhel vytočení kol $\pm 40°$
- servo Futaba S3010
- vývojový kit FRDM-K64F - specifikace na webu [9]
  - procesor ARM® Cortex®-M4 s frekvencí 120 MHz
  - 1 MB flash paměti a 256 KB SRAM
  - Ethernet
  - USB
  - SDHC
- baterie 7.2 V, 3000 mAh, NiMH
- wifi modul ESP8266 (pouze pro vývojové účely, během závodu odstraněn)
- Freescale Line Scan Camera Board - specifikace na webu [10]
- rozšiřující deska Freescale FRDM-TFC (na obrázku 2) připojena k desce pomocí GPIO portu - specifikace na webu [11]
  - 2 kanálový Motor Driver ICs (MC33887APVW)
  - 2 kanálové servo výstupy
  - duální interface pro jednořádkovou kameru

  - 2 vstupy pro rychlostní senzory
  - 2 potenciometry pro uživatelské funkce
  - 2 tlačítka pro uživatelské funkce
  - 4 polohové DIP přepínače pro uživatelské funkce
  - 4 LED diody (zelené) pro indikaci baterie nebo uživatelské funkce

2.1.2 Verze auta s rozšiřujícím Raspberry Pi

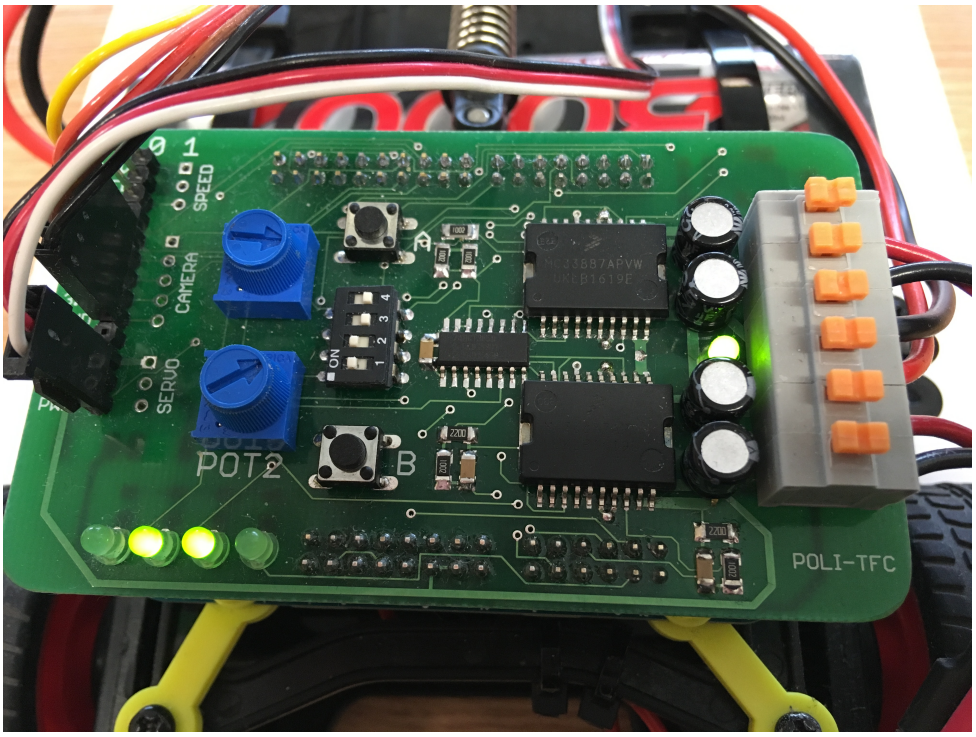
Vybavení auta pro účely bakalářské práce (obrázek 1b):

- skeleton auta - stejný, jako u auta určeného pro soutěž, kromě umístění kamery
  - výška umístění kamery od země 270 mm
  - vzdálenost umístění vztyčných ramen kamery: 40 mm před přední nápravou
- vývojový kit FRDM-KL25Z - specifikace na webu [12]
- baterie 7.2 V, 3000 mAh, NiMH
- rozšiřující deska Freescale FRDM-TFC totožná jako u verze auta pro závod
- Raspberry Pi 3 Model B V1.2 [16] připojeno ke kitu pomocí USB
  - 1.2GHz 64-bit quad-core ARMv8 CPU
  - 802.11n Wireless LAN
  - Bluetooth 4.1
  - 1GB RAM
  - 4 USB porty
  - 40 GPIO pins
  - Full HDMI port
  - Ethernet port
  - Camera interface (CSI)
  - Display interface (DSI)
  - 3.5mm audio jack
  - Micro SD card slot
  - VideoCore IV 3D graphics core
- Freescale Line Scan Camera Board - specifikace na webu [10]



(a) Soutěžní verze auta (b) Rozšířená verze auta

Obrázek 1: Fotografie robotických aut



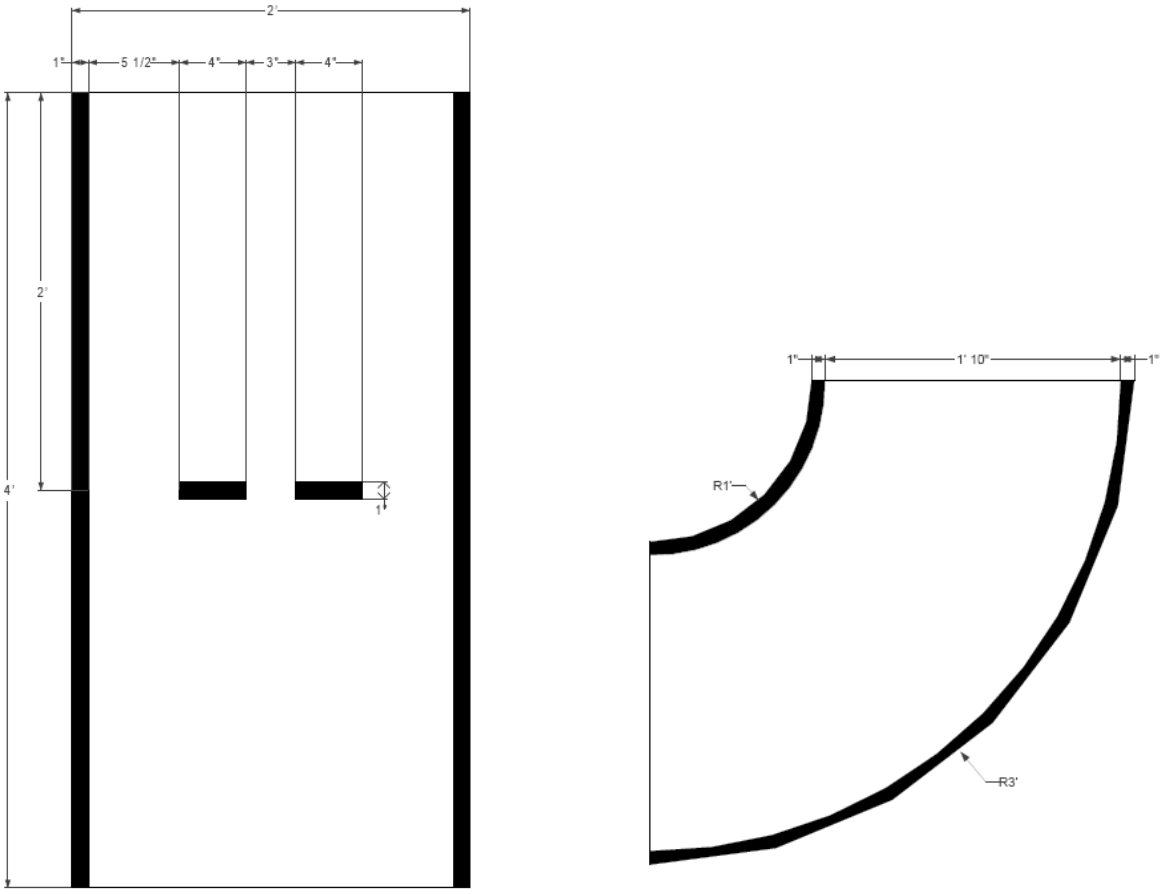
Obrázek 2: Rozšiřující deska podle specifikace Freescale FRDM-TFC

2.2 Dráha

Základní parametry testovací závodní dráhy - převzato z [3]:

- šířka dráhy ~600 mm
- povrch dráhy matně bílý s nepřerušovanou černou čarou o šířce 25 mm na každé straně trati

- závodní dráha může být protnuta pod úhlem 90° jinou částí dráhy - křižovatka
- závodní dráha může obsahovat stoupání, klesání a tunely



(a) Cílová čára

(b) Zatáčka

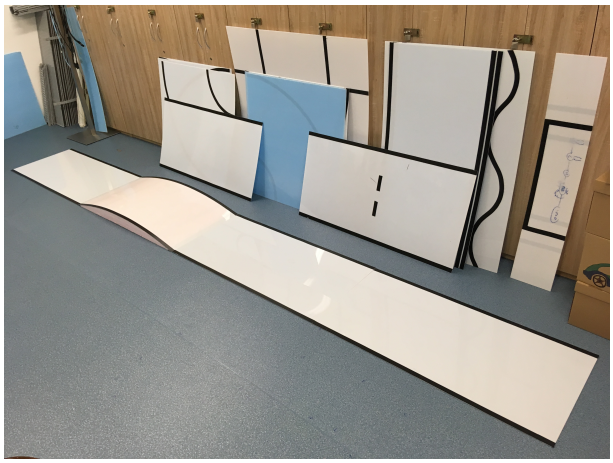
Obrázek 3: Ukázka předepsaných rozměrů panelů dráhy - převzato z [1]

Na obrázcích 3a a 3b je zobrazena specifikace panelů dráhy cílové čáry a zatáčky. Materiály a rozměry základních částí dráhy jsou k nalezení na komunitním webu NXP [1]. V následujícím textu práce nebyla uvažována část dráhy obsahující tunel.

Testovací dráha při práci na projektu měla jiné vlastnosti, než dráha použitá oficiálními organizátory soutěže NXP Cup. Lišily se zejména odrazivostí, kde oficiální dráha vykazovala známky jen lehkých odlesků, které ovšem byly na rozdíl od testovací dráhy pouze lokálního charakteru. Testovací dráha byla pod září světel pokryta odlesky téměř plošně. Dalším rozdílem mezi testovací a oficiální drahou byl v přilnavosti povrchu. Na povrchu testovací dráhy, bylo možné částečně využívat nižší přilnavosti, ve prospěch prokluzu kol a díky tomu projíždění zatáček se smykem. Na oficiální dráze byla přilnavost povrchu znatelně vyšší a jakékoliv pokusy o smyk na tomto povrchu byly neúspěšné.

2.3 Testovací sestavy dráhy

Na obrázcích 4a a 4b se nacházejí sestavené dráhy, na kterých probíhaly měření. Dráha č. 1 na obrázku 4a účelově testuje chování auta na přímé dráze ve spojení s kopcem a je užita pro testování s nízkými až maximálními rychlostmi. Dráha č. 2 byla sestavena za účelem testování maximální rychlosti auta a detekce cílové čáry. Naproti tomu dráha č. 3 na obrázku 4c je užita jako testovací prostředí pro širokou škálu testovaných vlastností auta, jako je zatáčení do obou stran, jízda na dráze s kopcem ve spojení se zatáčkou, průjezd křižovatkou a schopnost auta zastavit za cílovou čarou. Maximální rychlost auta na dráze č. 3 je omezována schopnostmi auta zvládat jednotlivé problematiky, především udržení stopy v zatáčkách.



(a) Testovací dráha č. 1

(b) Testovací dráha č. 2

(c) Testovací dráha č. 3

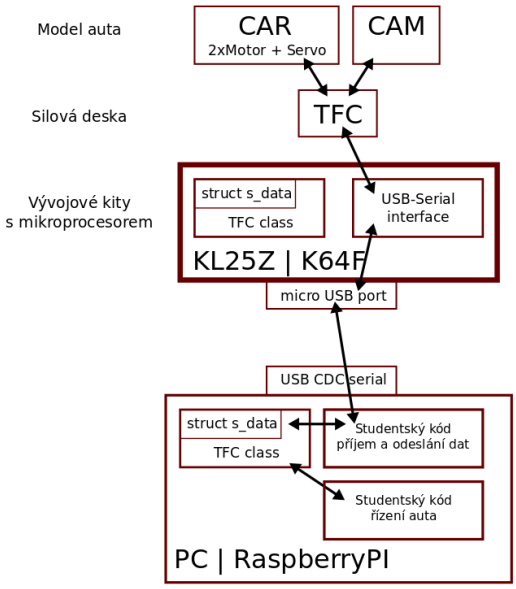
Obrázek 4: Fotografie testovacích drah užitých pro experimenty

3 Ovládání hardware auta

Společnost NXP dává vývojářům volně k dispozici pomocné knihovny pro práci s vývojovým kitem FRDM-K64F a FRDM-KL25Z. Tyto knihovny pracují s registry vývojového kitu. Na základě těchto dostupných knihoven výrobce, byl pro snadnější komunikaci s vývojovým kitem vytvořen zastřešující interface, pro komunikaci s oběma modely vývojového kitu, umožňující užívání více intuitivních funkcí pro komunikaci s hardware. Tento zastřešující interface, včetně jeho implementace, existuje v podobě třídy s názvem TFC. Interface TFC byl pro projekt, kterému náleží tento text, dodán panem Ing. Petrem Olivkou, Ph.D., který se podílel na přípravě týmů univerzity VŠB. Dodání existujícího interface se uskutečnilo z důvodu sjednocení přístupu k hardware všech závodních aut, která byla pro NXP Cup vyhrazena univerzitou.

Implementace interface TFC je závislá na použitém vývojovým kitu FRDM-K64F nebo FRDM-KL25Z. TFC implementace užívá knihoven poskytovaných výrobcem a umožňuje vývojáři práci se spojitými analogovými hodnotami, přičemž při posílání dat hardware jsou tyto hodnoty konvertovány na digitální nespojité hodnoty. V rámci A/D a D/A převodů samozřejmě dochází ke ztrátám informací, ale tyto ztráty jsou zanedbatelné.

Na obrázku 5 je náhled na schéma zapojení hardware a komunikace s TFC.



Obrázek 5: Schéma hardware a komunikace s TFC - převzato z [6]

Třída TFC definuje konstanty, se kterými bude dále v tomto textu pracováno.

Tabulka 1: Konstanty definované třídou TFC

Konstanta	**Hodnota**	**Popis**
CAMERA_LINE_LENGTH	128	Počet pixelů kamery
PWM_MINMAX	1000	-/+ rozsah pro PWM
SERVO_MINMAX	1000	-/+ rozsah pozic servomotoru
ADC_MAXVAL	0x0FFF	Max hodnota analogového vzorku z ADC
ANDATA_MINMAX	1000	-/+ rozsah pro analogové hodnoty
SERVO_DEFAULT_CENTER	1500	Střed pozice servomotoru - pulz v µs
SERVO_DEFAULT_MAX_LR	200	Defaultní -/+ rozsah šířky pulzu
SERVO_MAX_LR	400	Max -/+ povolený rozsah šířky pulzu
PWM_DEFAULT_MAX	200	Defaultní max -/+ PWM pracovního cyklu
PWM_MAX	500	Max povolené -/+ PWM pracovního cyklu
CMD_DATA	1	Příznak struktury s_data
CMD_SETTING	2	Příznak struktury s_setting
CMD_CONTROL	3	Příznak struktury s_control
STX	0x2	Počáteční řídící znak
ETX	0x3	Koncový řídící znak

Při komunikaci přes USB a Wi-Fi jsou užívány konstanty CMD_DATA, CMD_SETTING, CMD_CONTROL, STX a ETX.

Konstanty ADC_MAXVAL a ANDATA_MINMAX slouží jako maxima při A/D a D/A převodech.

Úroveň jasu dat z kamery je v rozmezí 0-0x0FFF, kdy 0 je černá a 0x0FFF je bílá.

PWM_MINMAX je -/+ konstanta hraničních hodnot rozsahu PWM obou motorů. Vůči této hodnotě je poměrově snížen výkon motorů pomocí konstant PWM_DEFAULT_MAX, která je nastavena defaultně při inicializaci, a PWM_MAX, která je dodatečně nastavena jako konečná poměrová hodnota PWM motorů. Je-li tedy nastaveno PWM_MAX na hodnotu 500, pak maximální výkon motorů bude nastaven na -/+ PWM_MINMAX / PWM_MAX, tedy 1000/500 = 1/2 výkon maxima motorů. PWM_MAX slouží jako omezovač výkonu, aby nebyl motor příliš velkým požadovaným výkonem poškozen. Při vývoji v aplikaci budou požadované hodnoty výkonu motoru tedy v rozsahu <-PWM_MINMAX, PWM_MINMAX>, ale metoda třídy TFC sníží ve výsledku poměrově tyto hodnoty na hodnoty v intervalu <-PWM_MINMAX * (PWM_MAX / PWM_MINMAX), PWM_MINMAX * (PWM_MAX / PWM_MINMAX)>. Kladné hodnoty požadovaného PWM představují jízdu vpřed, nulová hodnota zastavení motoru a záporné hodnoty jízdu auta vzad.

SERVO_DEFAULT_CENTER nastavuje hodnotu středu servomotoru pro natočení kol auta přímým směrem. Maximální hodnota vytáčení kol je dána intervalem <-SERVO_MINMAX, SERVO_MINMAX>, a tento interval je poměrově omezen konstantou SERVO_MAX_LR stejně, jako PWM motorů. Interval hodnot servomotoru je <-SERVO_MINMAX, SERVO_MINMAX>, ale hodnota vstupující do registrů je metodou

upravena na hodnoty intervalu <-SERVO_MINMAX * (SERVO_MAX_LR / SERVO_MINMAX), SERVO_MINMAX * (SERVO_MAX_LR / SERVO_MINMAX)>. Záporné hodnoty představují vytáčení kol doleva, nulová hodnota vytočení kol rovně a kladné hodnoty vytáčení kol doprava. SERVO_DEFAULT_MAX_LR představuje defaultní hodnotu pro maxima vytáčení kol doleva a doprava, pokud nebylo nastavení SERVO_MAX_LR užito k nastavení servomotoru. Při maximu 40% SERVO_MINMAX se kola již více mechanicky natočit nemohou. SERVO_MAX_LR slouží jako omezovač natáčení kol servomotorem, z důvodu ochrany hardware před mechanickým poškozením.

Detailní popis třídy TFC je popsán v dokumentaci dostupné na Redmine webu projektu [6].

4 Vzdálená komunikace s vývojovým kitem

Za účelem ladění, vizualizace dat a přenášení instrukcí z jiných zařízení do vývojových kitů FRDM-K64F a FRDM-KL25Z bylo užito prostředků periferií USB a modulů Wi-Fi.

4.1 Komunikace po sériové lince

Pro komunikaci kitů FRDM-KL25Z a FRDM-K64F se zařízením připojeným sériovou linkou (USB CDC), byla vytvořena verze řídícího programu hardware běžícího na kitu, která funguje na principu klient-server. V tomto vztahu klient-server vystupuje účastnické zařízení připojené sériovou linkou v roli klienta a na straně kitu vystupuje běžící program v roli serveru. Zařízení komunikují pomocí výměny struktur třídy TFC. Server komunikuje s klientem prostřednictvím struktury s_data a celý její obsah předává k dispozici klientovi. Tento předaný obsah struktury s_data slouží klientovi jako zdroj dat k vizualizaci na klientském zařízení nebo jako zdroj dat pro algoritmy určené k řízení auta. Server zpětně přijíma a zpracovává od klienta data ve formě struktur s_settings a s_control, pomocí kterých jsou ovládány jednotlivé motory auta a servo řídící natáčení kol.

Data jednotlivých struktur, jsou sériovou linkou ze zdrojového zařízení, posílána jako pole bytů (v binárním formátu little-endian) a na koncovém zařízení přijímána. Posílána data jsou zabalena do obálky pro snadné parsování dat po jejich přijetí na koncovém zařízení. Obálka je složena z počátečního řídícího znaku, délky posílané zprávy, typu zprávy, dat zprávy a koncového řídícího znaku. Složení obálky je zobrazeno v tabulce 2.

Tabulka 2: Obálka dat pro posílání zpráv po sériové lince

	STX	**Délka**		**CMD**	**Data konkrétní struktury**	**ETX**
počet bytů	1 byte	2 byty		1 byte	x bytů	1 byte
hodnota	0x02	LO byte	HI byte	0x01-0x03	data	0x03

CMD určuje typ cílové struktury a nabývá hodnot:

- 0x01 - Data,
- 0x02 - Setting,
- 0x03 - Control

U přijatého pole znaků je kontrolováno, jestli byla zpráva přijata celá pomocí nalezení existence počátečního STX a koncového ETX řídícího znaku v načtených datech[5]. Následně je hlavička a koncový řídící znak odebrán, zpráva je podle typu reinterpretována jako struktura s_data, s_control nebo s_setting a tato struktura je uložena do objektu třídy TFC.

Datová struktura s_data je posílána z auta každých 10 ms. Struktura s_control musí být autu opakovaně odesílána nejméně každou vteřinu - pokud auto neobdrží data 1 s, motory budou

vypnuty. Strukturu s_setting je autu potřeba poslat pouze pro inicializaci nastavení auta, a pak jen tehdy, je-li potřeba změnit nastavení. S přijatými daty struktur se dále pracuje pomocí třídy TFC.

Popis komunikace po sériové lince je uveden na webových stránkách wiki zdrojového projektu [6].

4.2 Komunikace po Wi-Fi

Za účelem bezdrátové komunikace bylo řešení rozšířeno o komunikaci pomocí modulu Wi-Fi. V případě komunikace po Wi-Fi přímo s vývojovým kitem FRDM-K64F a FRDM-KL25Z bylo využito externího Wi-Fi modulu ESP8266. V případě komunikace s deskou Raspberry Pi bylo užito integrovaného Wi-Fi modulu.

Za pomocí modulu ESP8266 bylo jednosměrně komunikováno s PC pomocí protokolu UDP za účelem zobrazování dat zpracovávaných řídícím programem. Posílanými daty byla struktura obsahující záznam z kamery, proudy motorů, PWM motorů, hodnotu natáčení servomotoru a timestamp. Odesílání paketu probíhá každých 10 ms, a protože frekvence snímání kamery je nastavena na 1 ms, je v paketu odesíláno 10 vzorků současně. Z důvodu omezení velikosti paketu na 20 kB maximální přenosovou rychlostí byl změnšen rozsah hodnot pole kamery z 0-0xFFF na 0-FF. Tento rozsah je pro účely zobrazování dat při ladění programu dostatečný.

Pro komunikaci Wi-Fi modulu integrovaného v Raspberry Pi bylo využito obousměrné komunikace pomocí TCP protokolu s PC. V Raspberry Pi běží v režimu démon program, který přijímá a přeposílá data mezi vývojovým kitem FRDM-K64F nebo FRDM-KL25Z a PC. Pomocí komunikace po USB stejným způsobem, jako je popsáno v kapitole 4.1, čte data z vývojového kitu a sockety je přeposílá přes Wi-Fi do PC. Program běžící v PC na základě získaných dat posílá zpět do Raspberry Pi řídící data - PWM jednotlivých motorů a hodnotu natočení servomotoru. Tyto hodnoty jsou následně zpět po USB předány ke zpracování vývojovému kitu.

5 Filtrační metody obrazu jednořádkové kamery

Základem pro schopnost auta chovat se autonomně je znalost pozice auta na dráze. Jelikož je prostředím pro provoz auta závodní dráha specifikovaná v kapitole 2.2, bude základní podmínkou znalost pozic krajnic dráhy. Je nutno od sebe s co nejlepším výsledkem odlišit plochu dráhy, krajnice, nečistoty a odlesky na dráze. To vše pouze za použití výstupu z jednořádkové kamery. Dráha je sama o sobě ideálním prostředím pro testování. Z toho důvodu, pokud bychom se pokusili přenést některou z metod filtrování obrazu, užitou v této práci, do dopravního provozu v reálném světě, nebyli bychom schopni s jejich pomocí automobil ovládat. Ovšem i na testovací dráze hrají svou významnou roli externí neovlivnitelné faktory. Těmi jsou

- zanesení dráhy nečistotami,
- intenzita, barva a úhel osvětlení v místě umístění závodní dráhy,
- stíny vržené na dráhu,
- a odlesky na dráze, které mohou vznikat působením osvětlení.

Navíc za plochou panelů dráhy se nachází podlaha místnosti, která může mít vlastnosti, se kterými musí být při hledání krajnic dráhy také počítáno - jako barva, vzor a odrazivost podlahy.

Černobílou jednořádkovou kameru (snímající úrovně jasu) o šířce 128 px bylo možné na konstrukci stojanu kamery polohovat vertikálně, a za pomocí tohoto polohování bylo možné nastavovat vzdálenost snímané plochy od auta. Protože vzdálenější snímaná plocha znamená širší záběr dráhy, bylo nutno najít nejvhodnější sklon kamery vzhledem k dráze. Na základě pozorování bylo zjištěno, že nejlepší vzdálenost snímané plochy od kamery při jejím umístění ve výšce 270 mm od země je mezi 20-30 cm. Hodnoty blíže 20 cm znamenaly pohotovější reagování auta při průjezdu zatáčkou, ale hodnoty blíže 30 cm byly schopny snímat celou šířku dráhy v oblastech dráhy mimo zatáčku. Naproti tomu měly hodnoty blíže 30 cm v zatáčkách horší vliv na řízení, jelikož byla snímána dráha v příliš vzdálené oblasti a mohlo tak při vyšších rychlostech auta docházet k tomu, že již nebyla kamerou snímána dráha, ale podlaha za ní.

Při vývoji bylo k dispozici množství objektivů s různou ohniskovou vzdáleností a tvarem čočky, které se podrobily selekci na základě nejlepších výsledků z pohledu šířky záběru snímané plochy a velikosti kontrastu. Větší viditelná plocha je důležitá pro snímání větší části dráhy najednou a vyšší kontrast umožňuje lépe rozeznat tmavé a světlé objekty na snímané ploše.

V počáteční fázi projektu probíhalo testování s objektivem s malou šířkou záběru, která zachytila při snímané vzdálenosti ~20 cm před autem, vždy nanejvýše jednu krajnici dráhy. S předpokladem, že lze účinně zjišťovat pozici nejvýše jedné krajnice se implementovala základní část řízení auta. Bylo zjištěno, že se znalostí nejvýše jedné krajnice, nelze účinně udržet auto v plynulém směru jízdy, za předpokladu, že chceme auto držet v okolí středu dráhy. Auto v takovém případě oscilovalo od jedné krajnice ke druhé. Auto by bylo možné držet v určité vzdálenosti od krajnice, ale nikdy by nebylo nedosaženo správného odhadu středu dráhy. Znalost středu dráhy

je důležitá pro plynulou jízdu po rovných částech dráhy. Pokud není střed dráhy znám a auto není k tomuto středu průběžně vedeno, může během jízdy při vysokých rychlostech dojít k tak velikému nárustu oscilace zatáčení, že auto vyjede z dráhy, i přesto, že jelo po rovné dráze.

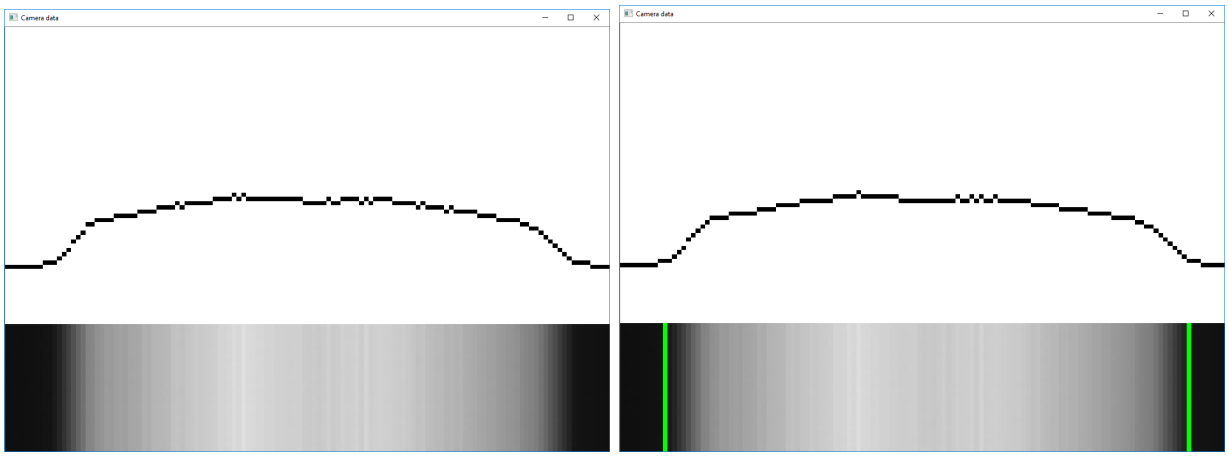
Z toho důvodu byl vybrán nový objektiv s větším záběrem snímané plochy, tzv. rybí oko (ang. fisheye). Objektiv rybí oko je typ ultra širokého objektivu, který zakřivuje scénu pro vytvoření hemisférického nebo široko panoramatického snímku [2]. S tímto typem objektivu bylo možné při snímané vzdálenosti ~20 cm před autem, a při optimálních podmínkách, zachytit obě krajnice současně. Pokud je možné odhalit pozici obou krajnic, je také možné vypočítat střed dráhy, a je předcházeno problému s oscilací na rovné dráze. Díky tomu je možné po rovných částech dráhy vyvinout daleko vyšší rychlost s minimálním předpokladem kolize.

V následujících podkapitolách bude dále rozvinuta problematika užitých metod filtrování záznamu jednořádkové kamery.

5.1 Problémy kamery a jejich řešení

5.1.1 Slepé oblasti záznamu

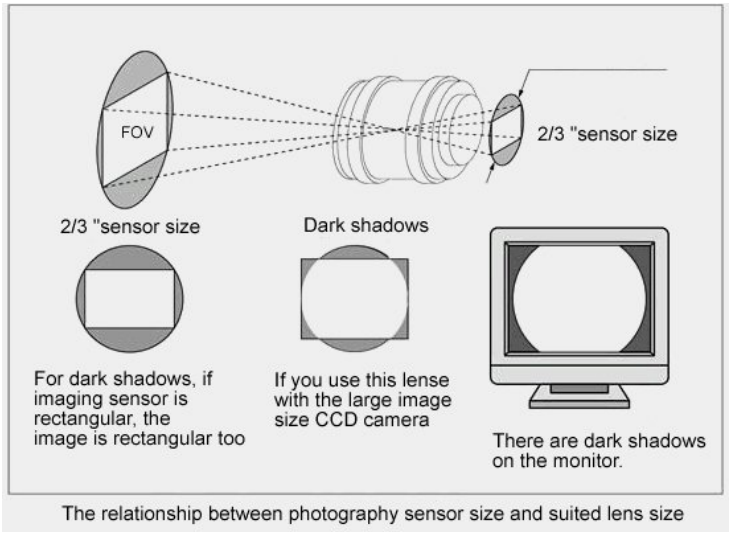
Než-li je však možné začít rozeznávávat objekty v obraze, je nutné zmínit problém snímaného záznamu, kdy se hodnoty počátečního a koncového indexu pole záznamu projevily jako nepoužitelné pro vyhodnocování obsahu záznamu. Na těchto indexech se nenachází skutečně snímaná dráha, ale jedná se o slepé místa kamery a na záznamu se projeví jako černá plocha. Tento problém lze dobře vidět při snímání bílé plochy bez jiných objektů v záznamu, viz obrázek 6a.



(a) Data z kamery při snímání bílé plochy (b) Nalezené indexy nepoužitelných částí záznamu

Obrázek 6: Detekce nepoužitelných částí záznamu

Důvodem existence problému tmavých částí záznamu je užití CCD senzoru většího než je plocha velikosti objektivu [4]. Tak dochází k osvícení pouze těch bodů CCD senzoru, kde je osazen objektiv a zbývající body senzoru nejsou prosvíceny a vznikají na snímku tmavé části. Vizualizaci tohoto efektu lze vidět na obrázku 7.



Obrázek 7: Vizualizace efektu objektivu a většího CCD senzoru. Převzato z [4]

Pro další práci se záznamem je potřeba tento efekt odstranit nebo vymezit hranice snímku, kde se nachází použitelná část snímku. Pro nalezení těchto indexů musíme nejdříve rozlišit, zda je úroveň jasu konkrétního bodu nad nebo pod úrovní, kterou považujeme za použitelnou. Vhodná metoda pro rozlišení těchto bodů je prahování popsané v kapitole 5.1.2. Pro detekci těchto bodů je potřeba nasměrovat kameru na čistě bílou plochu. Po aplikaci prahování postačí vyhledat ze středu záznamu doleva i doprava první výskyt bodu s hodnotou jasu nižší, než je určený práh. Indexy těchto nalezených bodů jsou hraničními indexy intervalu, na kterém se nachází použitelná data pro další zpracování. Viz obr 6b.

Aby bylo možné identifikovat krajnice v záznamu, ať už filtrovaného nebo originálního, je nutné omezit interval hledání na otevřený interval mezi nalezenými indexy (i0, i1). Na tomto omezeném intervalu lze aplikovat metody filtrování a hledat objekty krajnic a dráhy.

5.1.2 Vinětace

Každý z užitých objektivů se setkal s problémem zvaným efekt vinětace. Tento efekt je vadou optiky a způsobuje, že body na obou krajních indexech záznamu (na intervalu (i0, i1) zmíněném v kapitole 5.1.1) mají nižší jas než body na středních indexech záznamu. Střední body záznamu mají ovšem správnou hodnotu jasu oproti krajním bodům, tedy za předpokladu, že úroveň jasu snímané plochy je na celé šířce záznamu stejná. Každá optická soustava má tuto vadu unikátní, proto pro každou soustavu musí proběhnout kroky ke korekci jejího efektu vinětace znovu.

Vinětace je poměrně běžná optická vada objektivu a setkávám se s ní především u širokoúhlých objektivů a zoomů. Jedná se v podstatě o nestejnoměrné rozložení světla na snímku. To se projevuje tmavnutím rohů obrazu. Vinětace je způsobena jednak optickými zákonitostmi, ale také konstrukcí objektivu, hlavně jeho stavební délkou. Obecně se dá říci, že čím více čoček obsahuje objektiv, tím může být náchylnější k vinětaci. U širokoúhlých objektivů dochází k vinětaci

díky použití čoček s velkým zakřivením, světlo nemusí propouštět rovnoměrně. Vinětovat ale mohou například i objímka objektivu a nebo nesprávně zvolená sluneční clona, která zasahuje díky širokémi úhlu záběru do zorného pole objektivu [17].

Na obrázku 6a je dobře vidět problém s efektem vinětace, kde byl kamerou snímán čistě bílý povrch. Úroveň jasu jednotlivých obrazových bodů na záznamu stoupá a přechází v "oblouk", a za středem záznamu jas opět klesá.

Tuto vadu lze do větší míry odstranit za pomocí jednoduché matematiky. Je předpokládáno, že obrazový bod záznamu na indexu 63 nebo 64, který je středním indexem záznamu kamery o šířce 128 obrazových bodů, obsahuje správnou úroveň jasu záznamu. Této úrovně jasu by měl nabývat celý záznam, a tato hodnota bude zvolena jako výchozí hodnota jasu, ke které budou upraveny jasy ostatních obrazových bodů záznamu na intervalu (i0, i1). Poměrem požadovaného jasu a jasu obrazového bodu v záznamu bude získána hodnota násobku, kterou musí být obrazový bod vynásoben, aby byl efekt vinětace bodu opraven. Z toho vychází výpočet korekcni_nasobek[i] = pozadovany_jas/zaznam[i].

Pro každý bod záznamu uložíme jeho násobek, díky kterému jsme schopni upravit úroveň jasu bodu do výchozí hodnoty. Nastavení hodnot násobků by mělo proběhnout vždy pouze jednou jako počáteční konfigurace filtrace. Podmínkou ovšem je, že s nastavením objetivu kamery nebude během užívání prvotní konfigurace dále manipulováno. Posledním krokem pro opravu efektu vinětace je přenásobení provozních záznamů obrazu hodnotami vypočtených násobků - každý s odpovídajícím indexem.

```
int image[128] = { data };
for(int i = 0; i < 128; i++)
{
   image[i] = image[i] * correctionMultiplier[i];
}
```

Výpis 1: Korekce efektu vinětace

Těmito kroky byla odstraněna vada efektu vinětace. Výsledek korekce je zobrazen na obrázku 8a. Dále na obrázku 8b můžeme vidět (zleva) část dráhy, pravou krajnici a podlahu po korekci vinětace.

5.2 Prahování

Aby bylo možné identifikovat pozici krajnic ze záznamu kamery, obsahující úrovně jasu na šířce 128 px, omezeném otevřeným intervalem (i0, i1), lze použít jednu ze základních metod segmentace obrazu, tzv. prahování (anglicky tresholding).

Segmentace pomocí prahování je jednou z nejstarších metod segmentace obrazu na objekty. Díky výpočetní náročnosti jde o jednu z nejrychlejších metod segmentace. Metodu používáme tehdy, když se hledané objekty výrazně odlišují od pozadí. Volba prahu je pak stanovena tak, aby

(a) Bílá plocha po korekci vinětace

(b) Zleva: část dráhy, pravá krajnice a podlaha po korekci vinětace

Obrázek 8: Opravený efekt vinětace

došlo k oddělení požadovaných objektů od pozadí. Při prahování dochází k transformaci vstupního obrazu f na výstupní obraz g. Výstupní obraz nabývá hodnoty 1 pro objekty s intenzitou vetší než práh, v opacném případě 0. Protože g nabývá pouze hodnot 0 a 1, jedná se o binární obraz[18].

Toto jednoduché stavové přefiltrování dat napomáhá odhadnout (ale ne s jistotou lokalizovat) výskyt krajnice na záznamu dráhy. Jelikož ale testy probíhají v ideálním prostředí závodní dráhy, je možné předpokládat, že nalezené objekty pomocí prahování s hodnotou jasu nižší než je práh T, jsou objekty krajnic. Práh T je určen experimentálně nebo pomocí histogramu záznamu nalezením minima mezi dvěma maximálními vrcholy hodnot výskytu nejčastějších hodnot jasů v záznamu.

Postup pro výpočet prahu z histogramu bude následující:

- vstupem pro výpočet je záznam a počáteční a koncový index záznamu, na kterém se má funkce vykonat
- vytvoří se seznam uchovávající dvě hodnoty - hodnotu jasu a četnost nalezených bodů s tímto jasem
- projde se záznam od počátečního po koncový index a podle hodnoty jasu v bodě záznamu se naplní seznam
  - pokud v seznamu existuje hodnota jasu se zkoumanou hodnotou jasu bodu, četnost se iteruje
  - pokud v seznamu nebyla nalezena hodnota jasu, vytvoří se nový záznam v seznamu s hodnotou jasu zkoumaného bodu a s četností 1
- seřadí se záznamy v seznamu podle hodnoty jasu od nejmenšího po největší
- projde se celý seznam a je vyhledán index záznamu s největší četností

- projde se celý seznam a je vyhledán index záznamu s druhou největší četností
- projde se seznam od indexu záznamu s největší četností po index záznamu s druhou největší četností a je nalezena hodnota jasu záznamu s nejmenší četností
- hodnota jasu tohoto nalezeného minima četnosti mezi vrcholy maxima četnosti je hledaný práh jasu

Je-li užito prahování, jako hlavní filtrační metoda záznamu, je důležité, aby byla před samotným filtrováním aplikována korekce efektu vinětace (viz. kapitola 5.1.2). Pokud by prahování bylo užito bez korekce efektu vinětace, pak by mohlo dojít k nepřesnostem výsledků při určování hodnoty prahu nebo i při určování, zda bod patří pod nebo nad úroveň prahu, vinou zkreslení jasu záznamu na jeho okrajích.

Nalezení krajnic po aplikaci prahování je již velice snadnou záležitostí, budou-li uvažovány ideální podmínky - tj. na záznamu se nevyskytují nečistoty. Ze středu vyprahovaného záznamu bude vyhledána směrem doleva levá krajnice, která bude nalezena ve chvíli, kdy hodnota zkoumaného bodu je 0. Pokud žádný ze zkoumaných bodů při vyhledávání levé krajnice směrem doleva ze středu záznamu nenabývá hodnoty 0, pak se předpokládá, že levá krajnice na snímaném záznamu neexistuje. Nalezený index vyhledané krajnice bude využit pro další výpočty při řešení problematiky ovládání vozidla. Analogicky bude stejným způsobem vyhledávána pravá krajnice směrem doprava od středu záznamu. Pokud některá z krajnic není nalezena, je za nenalezenou krajnici dosazena hodnota příslušného indexu nepoužitelné části záznamu.

5.3 Detekce hran pomocí hledání přechodů

Další užitou metodou hledání krajnic dráhy je hledání přechodů jasu. Tato metoda nevyžaduje korekci efektu vinětace, pouze znalost indexů nepoužitelných částí záznamu. Prvním krokem pro aplikaci této metody je výpočet rozdílů jasu mezi jednotlivými obrazovými body (výpis 2).

```
int image[128] = { data };
int differences[128];
for(int i = 0; i < 127; i++)
{
   differences[i] = image[i] - image[i+1];
}
differences[127] = differences[126];
```

Výpis 2: Výpočet hodnot pole rozdílů jasu

Dalším krokem je určení prahu T mezi hodnotami rozdílů jasu, za pomoci kterého je detekován jasový skok, podezřelý z existence krajnice. Použitelná část záznamu (oblast zájmu) je rozdělena do dvou částí od středního indexu záznamu kamery po nepoužitelné části záznamu. Říkejme jim levá část a pravá část. Levá část náleží intervalu <první index oblasti zájmu, index

středu záznamu>. Pravá část náleží intervalu <index středu záznamu, poslední index oblasti zájmu>.

Následně jsou v každé části detekovány hrany. Detekování je prováděno od indexu středu záznamu po druhou mezní hodnotu indexu každého z intervalů. Nastavení detekčního prahu pro tuto metodu bylo provedeno experimentálně, pomocí vyčtení hodnot z grafu rozdílů jasu. Hodnota tohoto prahu musí být dostatečně vysoká, aby bylo zřejmé, že se jedná o skutečný přechod mezi úrovněmi jasu dráhy a krajnicemi (z černé na bílou nebo naopak). Pro experimentální provoz byla nastavena hodnota detekčního prahu na úroveň 300. Pro levou stranu, v případě nalezení hodnoty rozdílu jasu nižší než je záporná hodnota detekčního prahu, je tato hodnota považována a uznána za levou krajnici dráhy. Analogicky pro pravou stranu, je-li hodnota rozdílu jasu vyšší než je kladná hodnota detekčního prahu, je tato hodnota považována a uznána za pravou krajnici dráhy.

```
//urceni prahu prechodu jasu
#define FILTER_LEVEL 300
//levy index nepouzitelne casti zaznamu
#define LEFT_LINE_MAX 16
//nalezeni leve krajnice
int findLeftLine(int * image)
{
   for (int i = CAMERA_LENGTH/2; i >= LEFT_LINE_MAX; i--)
   {
      if (image[i] < (-FILTER_LEVEL))
      {
         return i;
      }
   }
   return LEFT_LINE_MAX;
}

//pravy index nepouzitelne casti zaznamu
#define RIGHT_LINE_MAX 113
//nalezeni prave krajnice
int findRightLine(int * image)
{
   for (int i = (CAMERA_LENGTH/2) + 1; i < RIGHT_LINE_MAX; i++)
   {
      if (image[i] > FILTER_LEVEL)
      {
         return i;
      }
   }
   return RIGHT_LINE_MAX;
}
```

Výpis 3: Vyhledání krajnic při užití filtrace obrazu pomocí přechodů

Tato metoda však není příliš imunní proti nestandardním světelným podmínkám (např. lokálním odleskům na dráze). Ale pro užití v rámci soutěže NXP Cup je metoda dostačující, proto byla v práci zachována.

6 Řízení auta

6.1 Základní řízení pomocí krajnic

Řízení pomocí krajnic využívá polohu krajnic dráhy a středu dráhy v aktuálním záznamu kamery. Pomocí pozic krajnic je vypočítán předpokládaný střed dráhy. Pokud není některá z krajnic nalezena, je její index nahrazen krajním indexem nepoužitelné části záznamu náležící chybějící straně záznamu. Tím je dosaženo vlastnosti, že i za předpokladu chybějící jedné nebo obou krajnic v záznamu je auto směřováno předpokládaným středem dráhy.

V zatáčkách se tato vlastnost projevuje tak, že vypočítaný střed dráhy se nachází mezi skutečně nalezenou krajnicí a dosazeným krajním indexem nepoužitelné části záznamu. Z toho důvodu je při snímání dráhy ve vzdálenosti ~20 cm od kamery auto vedeno podél vnější krajnice zatáčky. Natáčení kol auta je omezeno intervalem <-SERVO_MINMAX, SERVO_MINMAX> definované třídou TFC, kde hodnota 0 znamená, že kola auta jsou natočená přímo. Hodnoty menší než 0 nátáčejí kola auta směrem doleva, hodnoty větší než 0 natáčejí kola auta směrem doprava. Podle krajních hodnot intervalu natáčení kol <-SERVO_MINMAX, SERVO_MINMAX> je odvozen koeficient natočení kol podle vzdálenosti středu kamery od středu dráhy. Jsou-li uvažovány indexy nepoužitelných částí záznamu o hodnotách 16 a 113, a položení například levé krajnice co nejblíže středu kamery, tedy například na index 63, pak vypočítaný střed dráhy bude (63 + 113) / 2 = 88. Střed dráhy 88 bude odečten od středu kamery 63, takže 63 - 88 = -25. Absolutní hodnota této vypočtené vzdálenosti rovna 25 je maximální hodnota vzdálenosti mezi středem kamery a středem dráhy, při uvažovaných indexech nepoužitelných částí záznamu 16 a 113.

Následně bude vypočtena hodnota natočení kol při středu kamery od středu dráhy ve vzdálenosti jednoho obrazového bodu na záznamu. Je-li maximální hodnota natočení kol dána hodnotou SERVO_MINMAX, a maximální vzdálenost mezi středem kamery a středem dráhy je 25, pak SERVO_MINMAX/25 je hodnotou natočení kol pro vzdálenost mezi středem kamery a středem dráhy 1. Byl tak vypočítán předpokládaný koeficient natáčení kol. Jelikož by ale k natočení kol na maximum došlo až v případě najetí středu kamery ke krajnici ve vzdálenosti 1, což by vedlo k vyjíždění auta z dráhy z důvodu nemožnosti dostatečného zatáčení, tak je potřeba koeficient natáčení zvětšit. Vynásobením koeficientu natáčení kol hodnotou 2 bude plného vytočení kol dosaženo již při vzdálenosti 12,5 (tedy 12 celočíselně) obrazových bodů vzdálenosti středu kamery a středu dráhy, a tak nedojde k vytáčení kol na maximum až při limitní blízkosti středu kamery a středu dráhy. Tak bylo dosaženo konečné hodnoty koeficientu natáčení kol.

Dále je vypočítána velikost a směr jednorozměrného vektoru na bodech středu kamery a vypočítaného středu dráhy při jízdě auta. Výpočet probíhá jako střed dráhy - střed kamery = vektor. Směr tohoto vektoru slouží k identifikaci směru zatáčení kol. Je-li vektor nulový, jsou kola natočena přímo. Je-li nenulový, je vektor přenásoben koeficientem natáčení kol, a tím je získána požadovaná hodnota zatočení kol. Protože ale může požadovaná hodnota natáčení kol díky

přenásobení koeficientu natáčení kol hodnotou 2 nabývat až hodnot 2*±SERVO_MINMAX, což je mimo možný interval natáčení kol <-SERVO_MINMAX, SERVO_MINMAX>, je potřeba hodnoty větší (menší) než je požadovaný interval zmenšit (zvětšit) na krajní hodnotu intervalu <-SERVO_MINMAX, SERVO_MINMAX>. Tato hodnota je posílána jako požadovaná hodnota natáčení kol třídě TFC.

```
//nalezeni indexu leve casti nepouzitelneho zaznamu
int leftMax = ...
//nalezeni indexu prave casti nepouzitelneho zaznamu
int rightMax = ...
//nalezeni indexu nepouzitelne casti zaznamu, ktery je blize stredu kamery
int closerMax = (CAMERA_CENTER - leftMax < rightMax - CAMERA_CENTER)?
    leftMax : rightMax;
//nalezeni indexu stredu drahy pri polozeni krajnice na index
//stredu zaznamu kamery a druhe krajnice na index closerMax
float turningCoefficient = (CAMERA_CENTER + closerMax) / 2;
//ziskani vzdalenosti vypocteneho stredu drahy a stredu zaznamu kamery
turningCoefficient = abs(CAMERA_CENTER - turningCoefficient);
//nalezeni koeficientu pro nataceni kol a jeho prenasobeni 2
//pro zvyseni ucinnosti nataceni kol
turningCoefficient = (SERVO_MINMAX / turningCoefficient) * 2;
//nalezeni stredu drahy
float pathCenter = (float)(rightLineIndex + leftLineIndex) / 2;
//nalezeni vzdalenosti mezi stredem zaznamu kamery
//a vypoctenym stredem drahy
float correction = CAMERA_CENTER - center;
//prenasobeni vzdalenosti stredu zaznamu kamery
//a vypocitaneho stredu drahy vypoctenym koeficientem nataceni kol
float turning = correction * turningCoefficient;
//omezeni pozadovanych hodnot nataceni kol na krajni hodnoty intervalu
//<-SERVO_MINMAX, SERVO_MINMAX>
if(turning > SERVO_MINMAX)
{
   turning = SERVO_MINMAX;
}
else if(turning < -SERVO_MINMAX)
{
   turning = -SERVO_MINMAX;
}
```

Výpis 4: Algoritmus výpočtu natáčení kol

Experiment

Za účelem zjištění kvality řízení je potřeba otestovat jízdní vlastnosti auta s užitím metod detekce krajnic, metod řízení a metod ovládání motorů. Pro libovolnou variaci metod filtrace

a řízení auta bude provedeno testování jízdy na dráze s 20 opakováními za různých rychlostí. Chování při testech na dráze bude slovně popsáno. Za opakování experimentu je považováno absolvování jízdní trati od jejího počátku až po její konec. Rychlost auta bude ovládána pomocí potenciometru.

Pokus č. 1

Nastavení:

detekce krajnic pomocí hledání přechodů, základní řízení, užití elektronického diferenciálu, vzdálenost snímané plochy od kamery ~20 cm, dráha č. 3, rychlost: 25%

Popis chování:

Auto při nastavené vstupní rychlosti 25% nedosahuje potřebné rychlosti k vyjetí kopce. V zatáčkách se drží u středu dráhy s mírným náklonem na vnitřní stranu zatáčky. Během průjezdu křižovatkou auto zůstává ve stejném směru jízdy, ve kterém bylo před vjezdem do křižovatky.

Pokus č. 2

Nastavení:

detekce krajnic pomocí hledání přechodů, základní řízení, užití elektronického diferenciálu, vzdálenost snímané plochy od kamery ~20 cm, dráha č. 3, rychlost 50%

Popis chování:

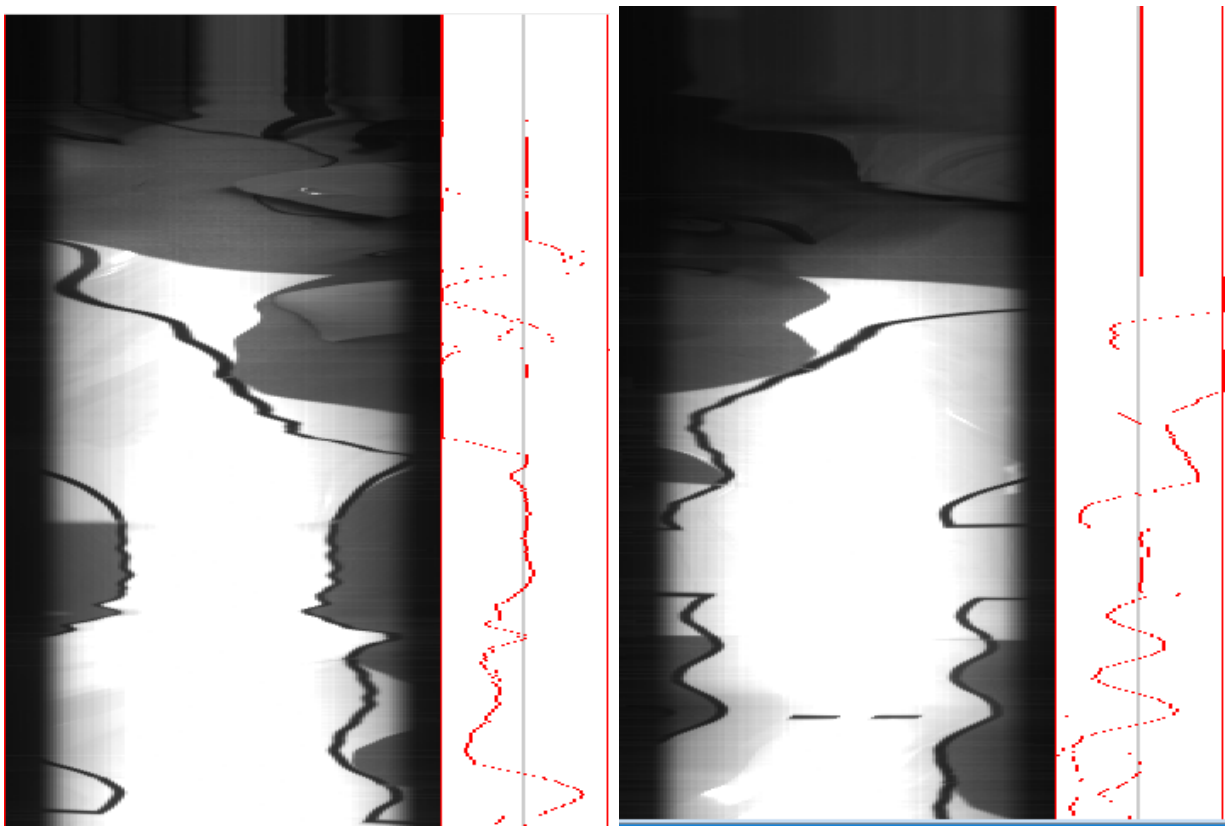
Auto dosahuje dostatečné rychlosti pro vyjetí kopce, po sjezdu z kopce je schopno včas reagovat a vytočit zatáčku. Při prvním pokusu auta o projetí tratě bez problémů projíždí celou trať, zatáčky projíždí podél vnější krajnice. Při opakovaném průjezdu tratí nastává opakovaně, ne však pravidelně, vyjetí z dráhy v zatáčce následující za křižovatkou.

Pro odhalení chyby je pořízen záznam z doby výjezdu z dráhy - obrázky 9a a 9b. Z pořízených záznamů z doby výjezdu z dráhy je vidět, že problém vzniká ve chvíli přejezdu krajnice přes střed kamery. V ten moment je algoritmem detekována blízká krajnice a tedy zatáčka v opačném směru než je skutečný směr projížděné zatáčky. Proto se auto začne od této krajnice vzdalovat opačným směrem a už není schopno návratu na dráhu.

6.2 Detekce cílové čáry

Jedním z požadavků soutěže NXP Cup je zastavení auta po projetí cílové čáry. Tento problém lze řešit dvěmi způsoby. Rozpoznáním objektu cílové čáry v historii záznamů. Nebo jednodušší metodou, kterou je nalezení vlastností záznamu, které se při každém projetí cílové čáry objeví, ale v ostatních záznamech se nevyskytují. V tomto textu bude řešena pouze zmíněná jednodušší metoda.

První myšlenka přichází s nalezením objektů čar cílové čáry za pomocí zjištění krajnic se vzájemnou vzdáleností daleko menší, než je obvyklá vzdálenost těchto krajnic. Problematickou se stává, při použití hledání krajnic pomocí hledání přechodů. Nachází-li se střed kamery na jedné z čar cílové čáry, pak první nalezená "krajnice"bude druhá z čar cílové čáry a druhá nalezená



(a) Záznam vyjetí z levotočivé zatáčky (b) Záznam vyjetí z pravotočivé zatáčky

Obrázek 9: Záznamy z vyjetí ze zatáčky

"krajnice"bude skutečná krajnice. Pozice těchto nalezených "krajnic"ale neodpovídají podmínce, která byla předpokládána pro nalezení cílové čáry: zjištění krajnic se vzájemnou vzdáleností je daleko menší, než je obvyklá vzdálenost těchto krajnic. Proto toto řešení není dostačující.

Jinou možností je nalezení dvou nejbližších černých čar na celé šířce záznamu a porovnání, zda tato vzdálenost mezi nimi je dostatečně malá, aby byla považována za detekovanou cílovou čáru. Postačujícím intervalem indexů záznamu pro vyhledávání bude <levá krajnice, pravá krajnice>.

Algoritmus pro vyhledání nejmenší vzdálenosti mezi všemi černými čarami na záznamu, využívající detekci krajnic pomocí hledání přechodů ve výpisu 5.

```
//prah prechodu jasu
#define FILTER_LEVEL 300
//hledana maximalni vzdalenosti mezi nalezenymi carami
#define FINISH_DISTANCE_BETWEEN_LINES 35
bool isFinishLineDetected(int leftLine, int rightLine, int *differences)
{
   int min = 127;
   //prvni zkoumana cara
   int checkedLineIndex = leftLine;
   for (int i = leftLine; i <= rightLine; i++)
   {
      //nalezeni prechodu s carou
      if (differences[i] > FILTER_LEVEL)
      {
         //tolerance 10 px mezi carami
         if ((i - checkedLineIndex < min) && i != leftLine
            && (i - checkedLineIndex > 10))
         {
            //vzdalenost zkoumane cary od predesle cary
            min = i - checkedLineIndex;
            checkedLineIndex = i;
            //cara nalezena, pokud je vzdalenost mezi carami
            //mensi nez FINISH_DISTANCE_BETWEEN_LINES
            if (min < FINISH_DISTANCE_BETWEEN_LINES)
            {
               return true;
            }
         }
      }
   }
   return false;
}
```

Výpis 5: Algoritmus hledání cílové čáry na poli rozdílů jasu

Experiment

Testování úspěšnosti algoritmu detekce cílové čáry při použití filtrace pomocí hledání přechodů. Rychlost jízdy auta bude nastavena na 250 a snímána vzdálenost plochy nastavena na ~30 cm před kamerou. Každá hodnota FINISH_DISTANCE_BETWEEN_LINES bude testována

právě 10x a její výsledky budou zaznamenány. Pokud auto zastaví mimo cílovou čáru, není dodatečné zastavení po průjezdu cílovou čarou bráno v potaz a zaznamenává se pouze chybné zastavení. Testování proběhne na dráze č. 2 (obrázek č. 4b).

Výstupem bude testovaná hodnota FINISH_DISTANCE_BETWEEN_LINES, počet úspěšných zastavení při průjezdu cílové čáry, počet chybných zastavení mimo oblast cílové čáry a počet projetí cílovou čarou bez zastavení. Na základě pokusu bude vyhodnocena optimální hodnota pro úspěšné projetí dráhy bez zastavení mimo oblast cílové čáry a pro správnou detekci cílové čáry. Také bude posouzena obecná funkčnost řešení detekce cílové čáry.

Výsledky měření v tabulce 3.

Tabulka 3: Měření úspěšnosti detekce cílové čáry

Vzdálenost [px]	**Nedetekováno**	**Detekováno**	**Nesprávné detekování**
15	5	5	0
17	6	2	2
18	3	5	2
20	0	7	3
20 (opakování)	0	7	3

Zhodnocení experimentu

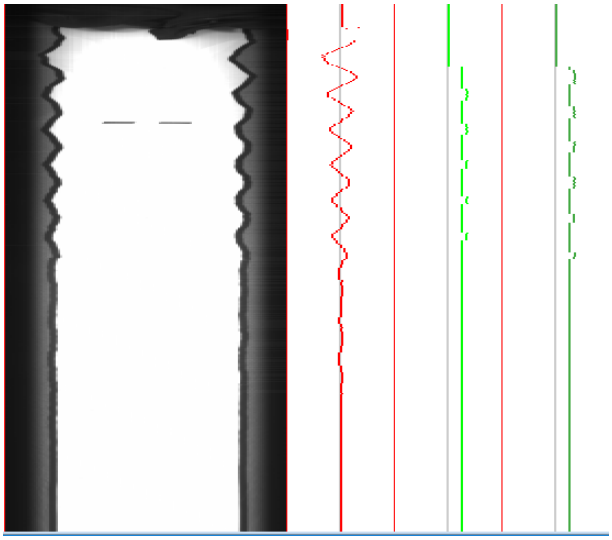
Z výsledků měření je zřetelné, že hodnoty menší než 18px jsou náchylné na časté projetí cílové čáry bez její správné detekce. Hodnota 18px byla méně náchylná na chybovost, ale ve třech případech také nedetekovala cílovou čáru, a proto byla testovaná vzdálenost mezi čarami opět navýšena. Hodnota 20px úspěšně detekovala 7 z 10 případů projetí cílové čáry a ve 3 případech zastavilo auto mimo cílovou čáru. V žádném z případů měřené hodnoty 20px nebyla vynechána detekce cílové čary. Pro pokus o zopakování výsledků s hodnotou 20px byl pokus s touto hodnotou zopakován a dle výsledku opakovaného pokusu byla úspěšnost potvrzena.

Zastavení mimo cílovou čáru bylo ve všech případech nesprávně detekováno v místě zatáčky za kopcem. V záznamu byla nesprávně detekována cílová čára mezi krajnicí a podlahou jejichž vzdálenost byla vyhodnocena jako menší než měřená hodnota. Tato chyba je přisuzována chybovosti detekce krajnice za pomoci hledání přechodů, a proto chybovost detekce cílové čáry v experimentu není dostatečně průkazná pro vyvrácení funkčnosti metody. Proto je tato metoda považována za funkční s minimální úspěšností 70% při správné kalibraci vzdálenosti.

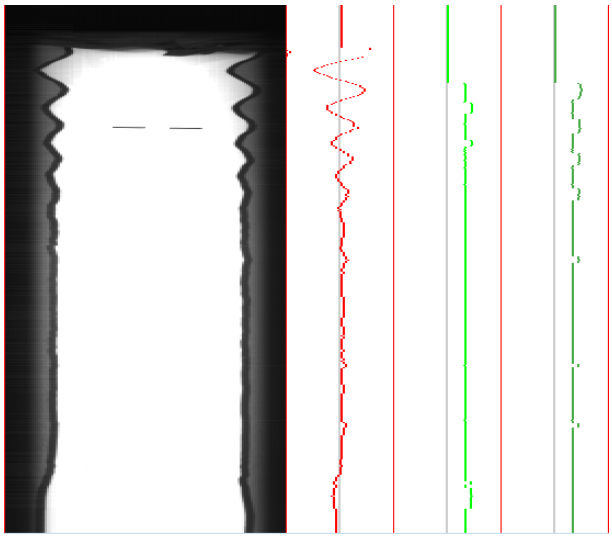
Experiment

Je potřeba vyzkoušet účinnost algoritmu ve vyšších rychlostech. Za účelem zjištění, zda je auto schopno detekovat cílovou čáru za pomocí užitého algoritmu v různých rychlostech, je potřeba užít dráhu bez zatáček a s co nejdelší rozjezdovou plochou. Pro tyto účely byla užita dráha č. 2 na obrázku 4b. Bude testována účinnost algoritmu za použití ovládání rychlosti pomocí potenciometru na rozšiřující desce auta, při nastavené hodnotě PWM_MAX 500, a s hodnotami potenciometru <0, 1000> reprezentující požadované PWM motorů (vysvětleno v kapitole 3).

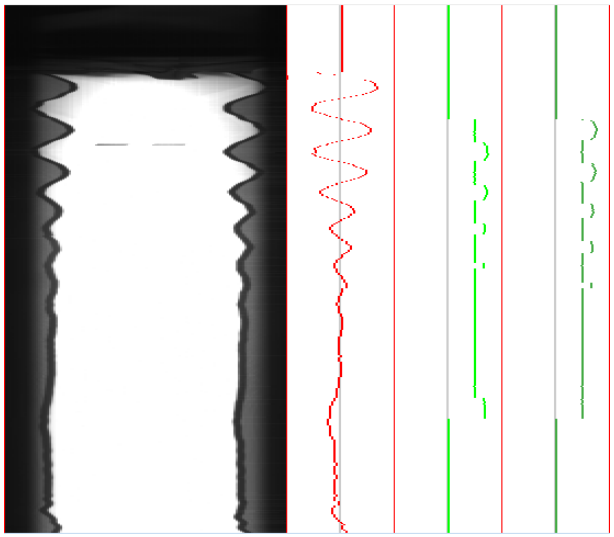
Rychlosti budou následující: 50%, 75% a 100% hodnoty nastavené potenciometrem. Tyto hodnoty odpovídají rychlostem 250, 375 a 500 v přepočtu na hodnoty PWM_MINMAX. Výstupem experimentu bude zhodnocení účinnosti algoritmu při daných rychlostech a záběry kamery, hodnoty natáčení kol a hodnoty vstupních rychlostí jednotlivých motorů.



(a) 50% rychlosti



(b) 75% rychlosti



(c) 100% rychlosti

Obrázek 10: Záznamy detekce cílové čáry v různých rychlostech auta

Na obrázcích 10a, 10b a 10c se nachází (zleva) nefiltrovaná sekvence snímků jízdy z kamery, sloupec s vizualizací hodnot natáčení kol, vstupní hodnota rychlosti levého motoru a vstupní hodnota rychlosti pravého motoru.

Z pozorování při experimentu bylo zřejmé, že ve všech případech nastavené vstupní rychlosti je auto schopno zaznamenat alespoň jeden snímek s hledanými vlastnostmi čáry. Zároveň je ale

znatelné, že se zvyšováním rychlosti dochází ke stále méně častému zaznamenávání snímků s cílovou čarou.

6.3 Řešení jízdy v zatáčkách

Problematikou jízdy auta při průjezdu zatáčkou je v reálném světě možnost smyku auta při vyšších rychlostech. To je způsobeno tím, že při průjezdu zatáčkou má kolo blíže ke středu otáčení menší obvodovou rychlost (kratší ujetá dráha a menší otáčky) a kolo dále od středu otáčení větší obvodovou rychlost (delší ujetá dráha a vyšší otáčky)[13]. Pokud ovšem mají obě kola při průjezdu zatáčkou stejné otáčky, pak může dojít k situaci, kdy jedno nebo obě kola projedou zatáčku ve smyku. Tento problém se přenáší i na případ robotického auta NXP.

Existují řešení tototo problému, jako jsou mechanický diferenciál a elektronický diferenciál. V rámci projektu náležícího této práci bylo užito principu elektronického diferenciálu.

6.3.1 Elektronický diferenciál

Diferenciál upravuje otáčky hnaných kol podle úhlu zatočení řiditelné nápravy. Elektronické diferenciály vychází z principu, že při zatáčení se mění dráha, kterou kola musí projet. Proto se vyhodnocuje úhel natočení kol. Z tohoto úhlu se vypočítá poloměr zatáčení a upraví rychlost jednotlivých kol. [19]

Postačujícími vstupy pro řešení elektronického diferenciálu jsou rychlost otáček a natáčení kol. Jelikož otáčkoměr v základním vybavení robotického auta není k dispozici, může být vstup rychlosti otáček nahrazen dostupnou informací o požadovaném výkonu motorů jednotlivých kol. Natáčení kol je předem vypočítáno z pozic krajnic v obraze, které je popsáno v kapitole 6.1.

Algoritmus elektrického diferenciálu je prací spolupracovníka projektu, a proto zde nebude detailně rozebírán. Z důvodu částečné nedotáčivosti použitého řešení byla empiricky zjištěna hodnota násobku pro dodatečnou úpravu otáček vnějšího kola. Změnami této hodnoty bylo možné měnit chování auta z nedotáčivosti až po přetáčivost.

6.4 Problematika řízení v kopci

Součástí testovací dráhy auta je úsek kopce. Za běžných okolností je pro vyjetí kopce potřeba zvýšit požadovaný výkon motorů. Na auto působí celkem tři síly. Tíhová síla, reakční síla vozovky a tahová síla motoru [14]. Aby mohlo auto vyjet kopec, je potřeba vyvinout tahovou sílu motoru vyšší než jakou působí tíhová síla a reakční síla vozovky pod daným úhlem sklonu vozovky.

Naopak při sjíždění kopce je za běžných okolností vhodné jízdu přibrždovat. Při brždění působí na auto tři síly. Tíhová síla, reakční síla vozovky a brzdná síla při daném úhlu sklonu vozovky [15].

V základní výbavě auta ovšem není gyroskop pro určení sklonu dráhy.

Naskýtá se ovšem myšlenka detekce nájezdu na kopec a sjezdu z kopce pomocí dostupné informace - velikosti proudu tekoucího jednotlivými motory hnací nápravy. Pokud auto najede

na kopec, proud tekoucí motory při stejném požadovaném výkonu motoru začne narůstat. Pokud auto kopec sjíždí, proud tekoucí motory při stejném požadovaném výkonu motoru prudce klesne, dokud nenarazí na dráhu s rovným sklonem, kdy proud krátce naroste a poté se ustálí. Z těchto předpokladů by bylo možné zvýšit požadovaný výkon motoru při detekci nájezdu na kopec, a při následné detekci prudkého klesání proudu rychlost auta přibrzdit. Tento přístup by nezaručil stejnou rychlost jízdy při jízdě na úseku kopce, ale zaručil by, že kopec bude autem úspěšně vyjet i při předem dané jízdní rychlosti nižší, než minimální požadované rychlosti pro vyjetí kopce. A zároveň by bylo zaručeno, že kopec bude sjet s přibržďováním a auto se vyhne komplikacím s průjezdem dráhy ve vyšší rychlosti, které auto dostáhne nárůstem rychlosti v důsledku sjíždění kopce.

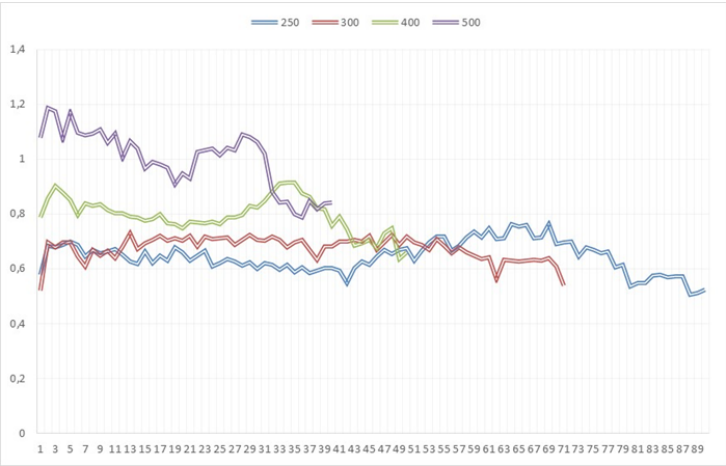
Experiment

Nalezení nejmenší rychlosti pro vyjetí kopce na dráze složené z rovných panelů dráhy a kopce (dráha č. 1, obrázek 4a). Během jízdy bude do souboru zaznamenávána hodnota proudů a z těchto hodnot bude vytvořen graf pro zjištění chování proudu při jízdě přes kopec. Dále bude zaznamenávána hodnota proudů při rychlostech vyšších, než je minimální rychlost pro vyjetí kopce, a bude z těchto hodnot vytvořen graf pro porovnání chování v různých rychlostech. Při tomto experimentu bude vypnutá úprava rychlosti jednotlivých motorů pomocí diferenciálu, a bude použita metoda filtrování pomocí přechodů.

Výsledky experimentu:

Při pokusu vyjíždění kopce byla nalezena nejmenší možná rychlost vyjetí kopce na hodnotě 250. Pokusy s rychlostí menší než 250 způsobily uváznutí auta v první polovině části kopce. Dále byly testovány rychlosti 300, 400 a 500.

Z grafu průměru proudů obou motorů lze vyčíst, že proud při vyjíždění kopce stoupá a při sjíždění kopce prudce klesá. Graf byl vytvořen při jízdě po rovném úseku dráhy následující kopcem a pokračující opět rovnou drahou (obrázek 11).



Obrázek 11: Graf průměrů proudů motorů na rovné dráze s kopcem

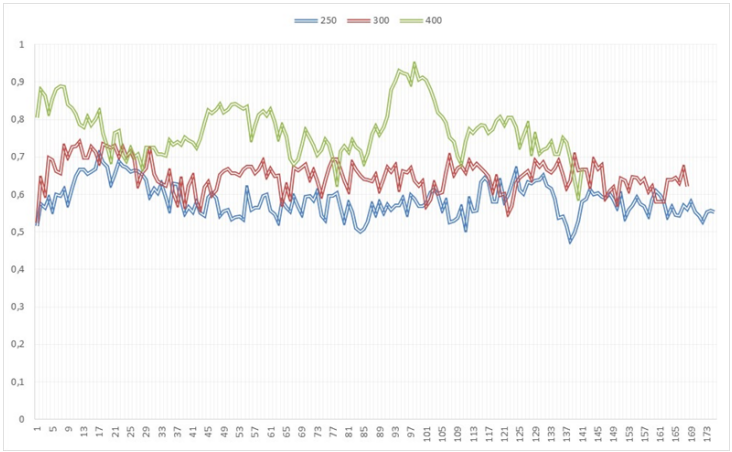
Experiment

Obměna předešlého experimentu na dráze obsahující kromě rovných částí a kopce i zatáčky (dráha č. 3, obrázek 4c).

Výsledky experimentu:

- Při rychlosti menší než 250 opět nebyl kopec vyjet a auto uvázlo v první polovině kopce.
- Při rychlosti 250 byl kopec úspěšně projet, včetně celé dráhy od startu po cíl ve směru od startu ke křižovatce.
- Při rychlosti 300 nebyl zaznamenán problém s vyjížděním kopce a dráha byla kompletně projeta.
- Při rychlosti 400 se auto chovalo podobně, jako při rychlosti 300.
- Při pokusech o překročení rychlosti 400 nebylo auto schopno projet bez diferenciálu a s filtrováním pomocí přechodů zatáčky, a proto byly z výsledků pokusu vynechány.

Nepředpokládaný problém nastává v úsecích dráhy, ve kterých auto projíždí zatáčkou. Bylo zjištěno, že při průjezdu zatáčky byl nárůst proudu podobný situaci najetí auta na kopec. A dále, že při opuštění zatáčky proud výrazně klesne, podobně, jako při sjíždění kopce. Toto lze pozorovat v grafu průměru proudů obou motorů při průjezdu celé dráhy včetně zatáček a kopce (obrázek 12).



Obrázek 12: Graf průměrů proudů motorů na dráze s kopcem

Tuto nežádanou detekci se nepodařilo eliminovat podmíněním, že detekce je platná pouze v případě natočení kol přímo, jelikož vyjíždění na kopec je možné i ze zatáčky, kdy jsou kola natočeny na některou ze stran. Navíc bylo zjištěno, že při rychlostech okolo hodnoty 300 není vždy patrný dostatečně vysoký nárůst proudu motorů.

Za pomoci těchto pozorování byla detekce vyjíždění a sjíždění kopce pouze pomocí velikostí proudů motorů vyhodnocena jako nedostatečná.

Aby byla detekce kopce prokazatelná, bylo by potřeba dodatečného senzoru gyroskopu. Řešení s gyroskopem není součástí této práce.

6.5 Pokročilé metody řízení

Mezi pokročilé metody řízení auta patří například naučené řízení za pomoci neuronových sítí, kterým je věnována kapitola 7.1.

7 Experimenty s modulem Raspberry Pi

7.1 Neuronové sítě - parametrická backpropagation

Jelikož vlastní řešení neuronových sítí ani jejich rozbor není cílem této práce, bude čtenář v následující části textu pouze obecně informován o základním principu neuronových sítí a backpropagation. Cílem této části textu je pouze demontrace experimentů s pokročilými metodami řízení, do kterých spadá i strojové učení. Pro rozsáhlejší informace o problematice neuronových sítí je doporučeno užít například zdrojové literatury této části textu. Část software vykonávající logiku backpropagation byl dodán panem Ing. Davidem Ježkem, Ph.D. za účelem demonstrace možného řešení.

Neuronové sítě jsou inspirovány biologickými neuronovými sítěmi. Tato vlastnost určitým způsobem předurčuje, že uměle vytvořené neuronové sítě by měly být schopny, z hlediska základních principů, se chovat stejně nebo alespoň podobně jako jejich biologické vzory.

Využívají distribuované, paralelní zpracování informace při provádění výpočtů. Jinými slovy ukládání, zpracování a předávání informace probíhá prostřednictvím celé neuronové sítě spíše než pomocí určitých paměťových míst. Tedy paměť a zpracování informace v neuronové síti je ve své přirozené podstatě spíše globální než lokální.

Znalosti jsou ukládány především prostřednictvím síly vazeb mezi jednotlivými neurony. Vazby mezi neurony vedoucí ke "správné odpovědi"jsou posilovány a naopak, vazby vedoucí k "špatné odpovědi"jsou oslabovány pomocí opakované expozice příkladů popisujících problémový prostor.

Učení je základní a podstatná vlastnost neuronových sítí.

Parametrická backpropagation využívá možnosti, aby byly podrobeny adaptaci nejen synaptické váhy, ale i prahy a strmosti sigmoidů. Taková síť se nazývá heterogenní a obecně v ní může mít každý neuron svou aktivační dynamiku 7.1.

Na základě vzorků v trénovací množině se síť učí reakce na vstupy. Pokud je pro řízení auta užita backpropagation, a na základě trénovací množiny nejsou reakce na vstupy odpovídající, pak je potřeba upravit trénovací množinu smazáním nadbytečných vzorků, které jsou podobné jiným vzorkům, ale s jinými požadovanými reakcemi. Pokud se takové vzorky v trénovací množině nenalézají, je možné, že je naopak potřeba dodat nový vzorek s požadovaným chováním. Pokud jsou k dispozici všechny potřebné vzorky, ale výstup neodpovídá požadovanému chování, pak je zřejmě potřeba upravit hodnoty koeficientů učení.

Experiment

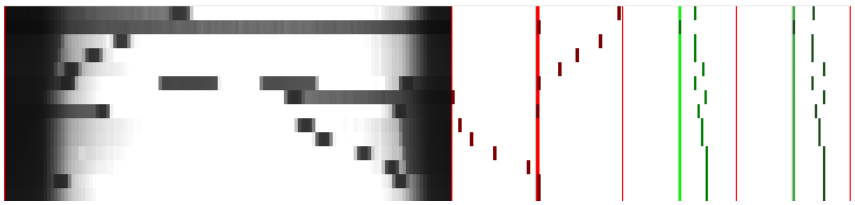
Je potřeba vyzkoušet řízení auta pomocí naučeného chování z parametrizovaného backpropagation. Pro experiment bude využit dodaný existující program pro zpracování logiky parametrizované backpropagation běžící na PC, komunikující s Raspberry Pi pomocí Wi-Fi, jak je

popsáno v kapitole 4.2, kde Raspberry Pi dále přeposílá řídící data vývojovému kitu po USB (kapitola 4.1).

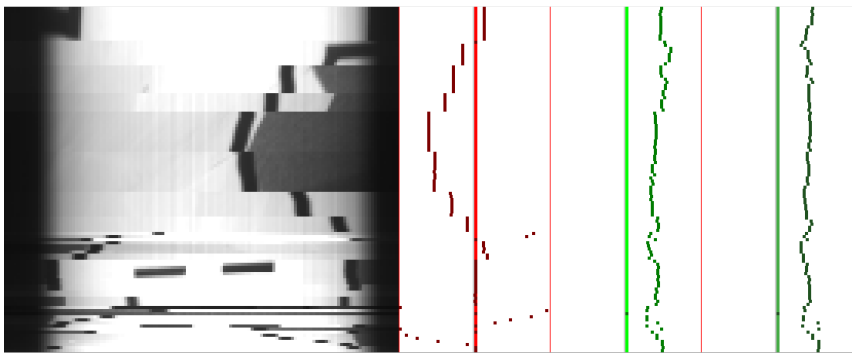
Nastavení sítě:

- Learning rate 0.3
- Lambda learning rate: 0.1
- Tresholding rate: 0.05
- Maximální chyba v procentech: 10%
- Počet neuronů vstupní vrstvy: 133 (128 záznam kamery, 2 PWM motoru, 2 proudy motorů a 1 hodnota serva)
- Počet skrytých vrstev: 1
- Počet neuronů ve skryté vrstvě: 20
- Počet výstupních neuronů: 3 (hodnota serva, PWM levého motoru, PWM pravého motoru)

Pro tento experiment byly použity dvě trénovací množiny při stejném nastavení koeficientů učení. Tyto trénovací množiny jsou na obrázcích 13 a 14 a každý řádek reprezentuje vzorek množiny. Sloupce obsahují (zleva) záznam z kamery, požadované natočení kol, PWM levého motoru a PWM pravého motoru.



Obrázek 13: Trénovací množina 1



Obrázek 14: Trénovací množina 2

Naučené chování pomocí trénovací množiny na obrázku 13, vykazovalo jízdní vlastnosti s oscilováním doprava a doleva i během průjezdu rovnou drahou, na křižovatce místo průjezdu rovně, opakovaně odbočovalo na pravou nebo levou stranu křižovatky.

Za účelem zlepšení jízdních vlastností byla rozšířena původní množina o nové vzorky, viz obrázek 14. Výsledkem rozšíření množiny, bylo více nepředvídatelné řízení auta, než u množiny s méně vzorky. Je pravděpodobné, že k nevhodnému chování došlo proto, že množina obsahovala dva podobné vzorky s velmi odlišnými požadovanými hodnotami servomotoru.

Na základě těchto experimentů, bylo zhodnoceno, že užití neuronových sítí pro řízení pomocí rozpoznávání obrazu není neúčinné, ale je především nepředvídatelné a vyžaduje spousty pokusů k nastavení požadovaného chování.

8 Soutěž NXP Cup

Soutěž původně pořádaná společností Freescale Semiconductor pod názvem Freescale Cup, později převzatou konkurenční firmou NXP, odkdy je soutěž dále vedena pod názvem NXP Cup. Soutěžícími jsou zástupci středních a vysokých škol z Evropy, Středního východu, případně Afriky. Je pořádána od roku 2011 a v době psaní této práce se konal již 6. ročník.

Týmy středních škol se účastní závodu na dráze s pouze jednou čarou, po které je auto vedeno. Týmy vysokých škol se účastní závodu na dráze ohraničené dvěmi černými čarami, jejichž vnitřkem je auto vedeno. Sestavy týmů musí být složeny pouze ze studentů s nejmenším počtem 2 studentů, a maximálně 3 studentů. Na sestavených drahách se mohou vyskytovat křižovatky, mosty, tunely a jiné předem nespecifikované panely dráhy.

Cílem soutěže je závod robotických aut zkonstruovaných a naprogramovaných týmy ke schopnostem autonomního řízení na dráze s předepsanými vlastnostmi a předem neurčeném uspořádání jejich částí. Hodnocena je rychlost jízdy, tedy čas, za který bylo schopno auto projet sestavenou dráhu. Na projetí dráhy má každý tým 3 pokusy a počítá se čas prvního celkového průjezdu drahou. V případě dráhy pro týmy vysoké školy, pokud auto vyjede více než třemi koly z dráhy ohraničené krajnicemi, je pokus považován za neúspěšný.

Soutěž probíhá ve fázích kvalifikačního kola a finálového kola. V případě umístění týmu mezi 3 nejlepší časy postupuje tým do finálového kola, kde se utká s dalšími postupujícími týmy z kvalifikačních kol konajících se v jiných zemích.

Kompletní pravidla soutěže jsou dostupná na Community webu NXP[7], případně konkrétní pravidla pro 6. ročník soutěže jsou dostupná v The NXP Cup 2016-2017 EMEA Rules[3].

8.1 Účast na soutěži

V rámci 6. ročníku soutěže se VŠB-TUO zúčastnilo se dvěma týmy. Prvním týmem zastupující VŠB-TUO byl "no-pistons", účastnící se již druhého ročníku soutěže. Druhým týmem byl tým auta s názvem "blind-driver", který je výsledkem projektu, ze kterého vychází tato práce.

Během testování dráhy, před započetím samotného závodu, bylo zjištěno, že oficiální dráha má jiné vlastnosti než dráha, na které byl projekt během vývoje testován - jako přilnavost auta na dráze a odrazivost materiálu. Odlesky na oficiální dráze byly lokálního charakteru s větší silou a přilnavost kol na dráze byla vyšší, proto nebylo auto schopné projíždět zatáčky s užitím smyku. V důsledku toho byly obnoveny pokusy s filtrační metodou obrazu prahování a bylo dodatečně upraveno nastavení hodnot elektrického diferenciálu.

Závodu se účastnilo 9 týmů vysokých škol a 4 týmy středních škol. Z vysokých škol dráhu úspěšně absolvovalo 6 týmů a ze středních škol všechny týmy. Sestava dráhy kvalifikačního kola týmů vysokých škol je k vidění na obrázku 15.

Tým "blind-driver"se umístil na 5. místě po úspěšném absolvování dráhy na první pokus, s časem 18 s. Tým "no-pistons"absolvoval dráhu také úspěšně s časem 14,8 s a celkovým umístěním v kvalifikačním kole na 4. místě.

Obrázek 15: Závodní dráha na soutěži - převzato z [8]

Nejlepšího času kola dráhy dosáhl vítězný tým kvalifikačního kola s 10,9 s.

9 Závěr

Cílem práce bylo navržení řešení autonomního řízení robotického auta NXP. Na začátku práce bylo popsáno dostupné vybavení a v následující části popsán přístup k jeho hardware při vývoji programu. Byla popsána komunikace s PC a s Raspberry Pi pomocí USB a Wi-Fi, které byly využívány k ladění programu a k experimentům s Raspberry Pi. Podrobně byly popsány užité filtrační metody obrazu a problémy optické soustavy, ke kterým byly taktéž navrženy řešení. Řízení robotického auta bylo řešeno metodami základního řízení pomocí nalezených krajnic dráhy v záznamu kamery, a pomocí učení neuronových sítí. Základní řízení se potýká s problémy při vyšších rychlostech s vyjížděním ze zatáčky. V navazujících pracích je potřeba se zaměřit na tento problém, a tím výrazně zvýšit kvalitu projektu jako celku. Byla popsána detekce cílové čáry, která i přes svou jednoduchost dosahuje dobrých výsledků při aplikaci. Jízda v zatáčkách pomocí elektronického diferenciálu byla obecně objasněna, ale nepopisována podrobně, jelikož byla vyvíjena výhradně jiným členem týmu závodu NXP. Pro detekci jízdy po kopci bylo navrženo možné řešení pomocí sledování proudů motorů, které se ale v rámci pokusů projevilo jako nedostatečné. Proto by v navazující práci měla být tato problematika dále řešena i za pomocí dodatečných senzorů. Byl popsán experiment provedený s naučenými jízdními vlastnostmi pomocí neuronových sítí, které se ukázalo jako zajímavý experiment, ale pro účely závodění není tato metoda příliš vhodná pro svou nepředvídatelnost. V poslední části byla popsána samotná účast na soutěži s výsledky závodního auta s řešením popsaným v tomto textu.

Literatura

[1] *NXP Cup Race Track Details* [online] [cit. 2017-04-18]. Dostupné z: https://community.nxp.com/docs/DOC-1092

[2] *Fisheye Lens Definition* [online] [cit. 2017-04-18]. Dostupné z: https://www.canon.com.au/explore/glossary/fisheye-lens

[3] *The NXP Cup 2016-2017 EMEA Rules* [online] [cit. 2017-04-27]. Dostupné z: https://community.nxp.com/servlet/JiveServlet/download/332303-2-386842/NXP%20Cup%20%202016-2017%20EMEA%20Challenge%20Rules.pdf

[4] *AZURE Photonics Co., Ltd-machine vision lenses & CCTV lenses, optical components, filters, prisms, lasers* [online] [cit. 2017-04-27]. Dostupné z: http://www.azurephotonics.com/?sn=casenr&id=12

[5] *Control characters in ASCII and Unicode* [online] [cit. 2017-04-27]. Dostupné z: http://www.aivosto.com/vbtips/control-characters.html

[6] *Wiki - NXP Cup Resources - REDMINE for SWI group* [online] [cit. 2017-04-26]. Dostupné z: https://rmine.cs.vsb.cz/projects/nxp-cup-resources/wiki/

[7] *NXP Community* [online] [cit. 2017-04-18]. Dostupné z: https://community.nxp.com/

[8] *NXP Cup - Qualifications Ostrava, Czech Republic* [online] [cit. 2017-04-27]. Dostupné z: https://community.nxp.com/docs/DOC-333972

[9] *FRDM-K64F* [online] [cit. 2017-04-27]. Dostupné z: https://developer.mbed.org/platforms/FRDM-K64F/

[10] *Line Scan Camera Board* [online] [cit. 2017-04-27]. Dostupné z: https://community.nxp.com/docs/DOC-1058

[11] *Freescale Cup Motor Driver Board* [online] [cit. 2017-04-27]. Dostupné z: https://community.nxp.com/docs/DOC-1059

[12] *mbed FRDM KL25Z* [online] [cit. 2017-04-27]. Dostupné z: https://developer.mbed.org/handbook/mbed-FRDM-KL25Z

[13] *Differential (mechanical device) - Wikipedia* [online] [cit. 2017-04-27]. Dostupné z: https://en.wikipedia.org/wiki/Differential_(mechanical_device)

[14] *Stoupající auto - Sbírka úloh* [online] [cit. 2017-04-27]. Dostupné z: http://reseneulohy.cz/216/stoupajici-auto

[15] *Nákladní auto a kopec - Sbírka úloh* [online] [cit. 2017-04-27]. Dostupné z: http://reseneulohy.cz/215/nakladni-auto-a-kopec

[16] *Raspberry Pi 3 Model B - Raspberry Pi* [online] [cit. 2017-04-27]. Dostupné z: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[17] *Vinětace - Megapixel* [online] [cit. 2017-04-27]. Dostupné z: https://www.megapixel.cz/vinetace

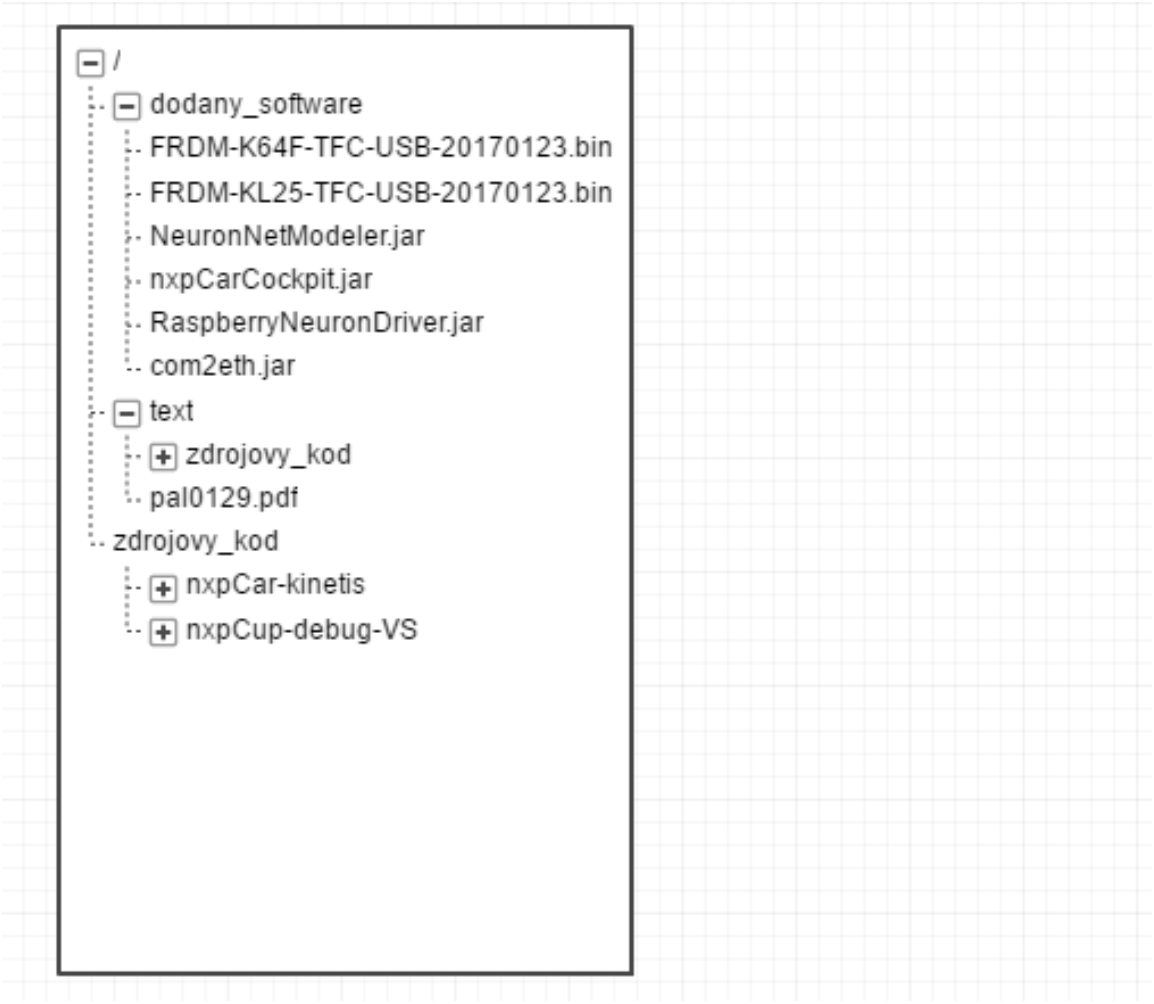
[18] *ZDO - cvičení 2 :: Katedra kybernetiky ZČU* [online] [cit. 2017-04-27]. Dostupné z: http://www.kky.zcu.cz/cs/courses/zdo/lesson2

[19] *TIM2 - Konstrukce VERZE 2 - Popis elektronického diferenciálu* [online] [cit. 2017-04-27]. Dostupné z: http://www.tim2.wz.cz/diferencial.php

[20] VONDRÁK, Ivo. *Umělá inteligence a neuronové sítě*. Dot. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, 2001. ISBN 80-7078-259-5.

A Přílohy na CD/DVD

Na obrázku 16 je zobrazena struktura obsahu přiloženého CD.



Obrázek 16: Strom obsahu CD

dodany_software

V této složce se nachází veškerý dodaný existující software užívaný během práce na projektu

- FRDM-K64F-TFC-USB-20170123.bin - binární soubor TFC klient-server pro USB pro nahrání do kitu auta FRDM-K64F
- FRDM-KL25-TFC-USB-20170123.bin - binární soubor TFC klient-server pro USB pro nahrání do kitu auta FRDM-KL25Z
- NeuronNetModeler.jar - program zpracovávající neuronové sítě

- nxpCarCockpit.jar - nástroj pro vizualizaci dat auta po USB a Wi-Fi na uživatelském PC
- RaspberryNeuronDriver.jar - program pro řízení auta pomocí dat z neuronové sítě
- com2eth.jar - program démon běžící na Raspberry Pi, přeposílající data mezi PC a kitem

text

- zdrojovy_kod - složka obsahující zdrojový kód latex této práce
- pal0129.pdf - PDF soubor této práce

zdrojovy_kod

- nxpCar-kinetis - složka obsahující projekt pro kinetis studio
- nxpCup-debug-VS - složka obsahující ladící verzi programu pro auto pro prostředí Visual Studio 2015
  - README.txt - soubor upozorňující na nutnost stáhnutí dodatečné knihovny OpenCV pro kompilaci programu